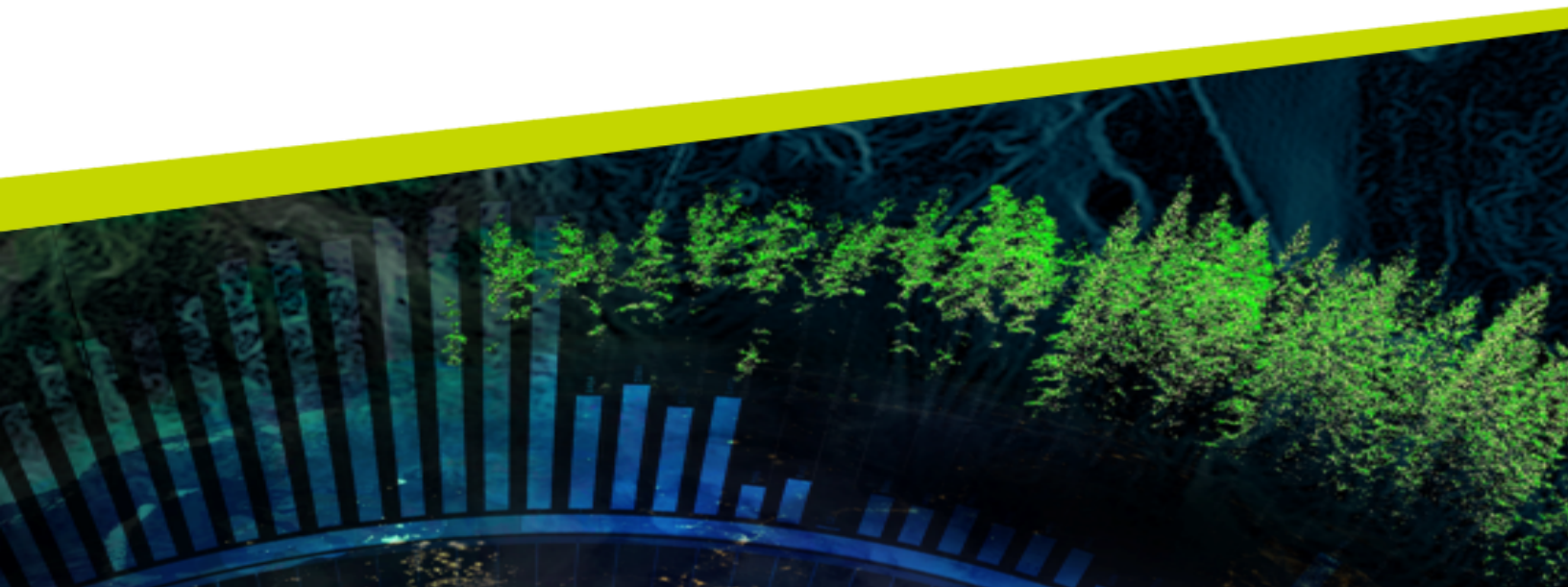
INFERENCIA Y MODELOS ESTADÍSTICOS

Jacqueline Köhler C. y José Luis Jara V.



CAPÍTULO 15. REGRESIÓN LINEAL MÚLTIPLE

En el capítulo anterior conocimos los principios detrás de la regresión lineal, considerando para ello una única variable predictora y una variable de respuesta. Sin embargo, en la vida real es más frecuente que un fenómeno sea explicado por muchas variables. En consecuencia, en este capítulo presentaremos un nuevo modelo lineal más complejo: la regresión lineal múltiple (RLM), correspondiente al caso de una única respuesta con múltiples predictores. Para ello tomaremos como base los textos de Field y col. (2012, pp. 245-311) y Diez y col. (2017, pp. 372-385).

Una regresión lineal con múltiples variables tiene la forma que se presenta en la ecuación 15.1, donde:

- Cada x_i es un predictor.
- Cada β_i corresponde a un parámetro del modelo.
- k es la cantidad de predictores.
- $\hat{y}$ es una estimación de la respuesta.

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + +\beta_k x_k \tag{15.1}$$

Una vez más, al ajustar el modelo mediante el método de mínimos cuadrados, buscamos minimizar la suma de los cuadrados de los residuos (ecuación 15.2), proceso que se vuelve más complejo a medida que aumenta la cantidad de variables por lo que suele hacerse mediante el uso de software.

$$\text{mín} \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{15.2}$$

Al igual que en el caso de la regresión lineal simple (RLS), la RLM requiere verificar algunas condiciones:

1. La distribución de los residuos debe ser cercana a la normal.
2. La variabilidad de los residuos debe ser aproximadamente constante.
3. Los residuos deben ser independientes entre sí.
4. Cada variable se relaciona linealmente con la respuesta.

Para los ejemplos de este capítulo usaremos una vez más el conjunto de datos `mtcars`, cuyas variables describimos en la tabla 14.1. Como punto de partida, recordemos la RLS para predecir el rendimiento de un automóvil (usando el mismo conjunto de datos) a partir de su peso y ajustada con todo el conjunto de entrenamiento (figura 14.7).

Ahora consideremos una RLM con dos predictores para el rendimiento: el peso (columna `wt`) y el tiempo mínimo requerido para recorrer un cuarto de milla (columna `qsec`), modelo que podemos obtener mediante el script 15.1 y que se muestra en la figura 15.1. El procedimiento para ajustar la RLM en R es el mismo que usamos en el capítulo anterior, pero ahora en el lado derecho de la fórmula para ajustar el modelo tenemos que combinar ambos predictores.

Script 15.1: regresión lineal para predecir el rendimiento de un automóvil a partir de dos variables.

```
library(scatterplot3d)

# Cargar los datos.
datos <- mtcars

# Ajustar modelo usando validación cruzada de 5 pliegues.
modelo <- lm(mpg ~ wt + qsec, data = datos)
print(summary(modelo))

```

```
Call:
lm(formula = mpg ~ wt + qsec, data = datos)

Residuals:
    Min      1Q  Median      3Q     Max
-4.3962 -2.1431 -0.2129  1.4915  5.7486

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  19.7462     5.2521   3.760 0.000765 ***
wt           -5.0480     0.4840 -10.430 2.52e-11 ***
qsec          0.9292     0.2650   3.506 0.001500 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.596 on 29 degrees of freedom
Multiple R-squared:  0.8264,Adjusted R-squared:  0.8144
F-statistic: 69.03 on 2 and 29 DF,  p-value: 9.395e-12
```

Figura 15.1: descripción del modelo lineal para predecir el rendimiento de un automóvil a partir de dos variables.

```
# Graficar modelo ajustado.
g <- scatterplot3d(datos$wt, datos$qsec, datos$mpg, type = "p",
                   highlight.3d = TRUE, pch = 20, xlab = "Peso [lb x 1000]",
                   ylab = "Rendimiento [millas/galón]",
                   zlab = "1/4 de milla [s]")

g$plane3d(modelo ,draw_polygon = TRUE, draw_lines = TRUE)
print(g)
```

Para usar este modelo a fin de predecir valores para la respuesta a partir de un nuevo conjunto de datos, usamos una vez más la función `predict()`, del mismo modo que vimos en el capítulo 14 para la RLS.

Como en este caso tenemos dos predictores, lo que se ajusta ya no es una recta, sino un plano, como muestra la figura 15.2. Así, ya no tiene sentido hablar de la pendiente de la recta al momento de interpretar los parámetros del modelo. Un análisis de regresión lineal con múltiples variables busca aislar la relación entre cada predictor y la respuesta, por lo que el coeficiente β_i, asociado al i-ésimo predictor, representa el cambio esperado que se produce en la respuesta al incrementar dicho predictor en una unidad, **manteniendo constantes todos los demás predictores**. Si b_1 es el parámetro ajustado para el peso y b_2 es el parámetro ajustado para el cuarto de milla, b_1 puede entenderse como la pendiente del plano de la figura 15.1 con respecto al eje y, mientras que b_2 puede entenderse como la pendiente del mismo plano con respecto al eje z. A su vez, la intercepción fija la posición del plano con respecto al origen.

Desde luego, podemos extender esta idea para más de dos predictores. En tal caso ajustamos un hiperplano cuya forma y ubicación están dadas por los parámetros del modelo.

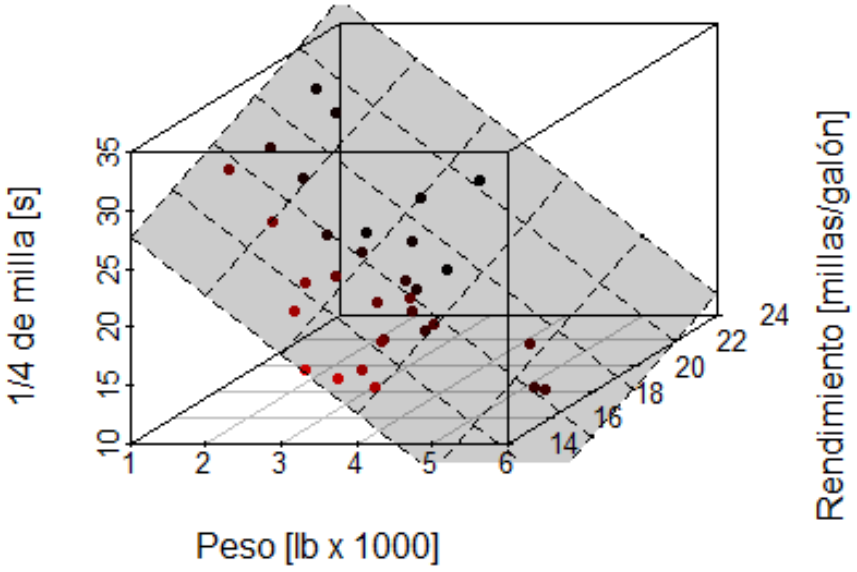


Figura 15.2: plano ajustado para la RLM con dos predictores.

15.1 RLM CON PREDICTORES CATEGÓRICOS

En el capítulo 14 habíamos señalado que para usar una variable categórica con dos niveles como predictor, esta debe ser transformada en una variable indicadora. Desde luego, podemos extender la misma idea para el caso de variables categóricas con más niveles.

Imaginemos que tenemos una variable categórica con k niveles. El primer paso consiste en crear $k-1$ nuevas variables artificiales. A continuación, para cada una de estas nuevas variables, escogemos un nivel diferente de la variable original y asignamos un 1 a todas las observaciones que tengan ese nivel y un 0 a las restantes. Para entender mejor esta idea, supongamos que un conjunto de datos tiene la variable categórica `tipo`, con cuatro niveles: `A`, `B`, `C` y `D`. Creamos tres nuevas variables, por ejemplo, `tipo_A`, `tipo_B` y `tipo_C`. La variable `tipo_A` contiene tantas observaciones como la variable original, con un 1 para aquellas observaciones en que `tipo` toma el valor `A` y un 0 para las restantes. Para las variables `tipo_B` y `tipo_C` se procede de manera análoga. Puesto que solo se crean $k-1$ variables artificiales, se descarta aquella correspondiente al nivel `D` de la variable `tipo`.

En R, podemos hacer esta tarea de manera sencilla mediante la función `dummy.data.frame(data, names, drop)` del paquete `dummies`, donde:

- `data`: matriz de datos.
- `names`: nombres de las columnas para las que se desea crear variables artificiales. Si se omite este argumento, se crean variables artificiales para todas las variables categóricas y de tipo string.
- `drop`: indicador booleano que, cuando es verdadero, descarta la variable original del resultado.

Es importante señalar que esta función crea una variable artificial por cada nivel de la variable categórica,

por lo que debemos descartar una de ellas.

El script 15.2 muestra el funcionamiento de `dummy.data.frame()` para una matriz de datos con dos variables categóricas. La tabla 15.1 muestra, en las columnas de la izquierda, el conjunto de datos original y en las de la derecha, el conjunto de datos con las nuevas variables artificiales.

sujeto	sexo	tipo	valor
1	F	B	1.68
2	F	D	2.79
3	M	A	1.92
4	M	B	2.26
5	M	A	2.10
6	M	C	2.63
7	F	D	2.19
8	M	D	3.62
9	F	D	2.76
10	F	A	1.26

sujeto	sexoM	tipoB	tipoC	tipoD	valor
1	0	1	0	0	1.68
2	0	0	0	1	2.79
3	1	0	0	0	1.92
4	1	1	0	0	2.26
5	1	0	0	0	2.10
6	1	0	1	0	2.63
7	0	0	0	1	2.19
8	1	0	0	1	3.62
9	0	0	0	1	2.76
10	0	0	0	0	1.26

Tabla 15.1: creación de variables artificiales para una matriz de datos con variables categóricas.

Script 15.2: creación de variables artificiales para variables categóricas.

```
library(dummies)

# Crear una matrz de datos.
sujeto <- 1:10
sexo <- c("F", "F", "M", "M", "M", "M", "F", "M", "F", "F")
tipo <- c("B", "D", "A", "B", "A", "C", "D", "D", "D", "A")
valor <- c(1.68, 2.79, 1.92, 2.26, 2.1, 2.63, 2.19, 3.62, 2.76, 1.26)
datos <- data.frame(sujeto , sexo , tipo, valor)

# Crear variables artificiales.
datos.dummy <- dummy.data.frame(datos , drop = TRUE)
datos.dummy[["sexoF"]] <- NULL
datos.dummy[["tipoA"]] <- NULL

# Crear modelos lineales.
m1 <- lm(valor ~ sexo + tipo, datos)
print(m1)

m2 <- lm(valor ~ sexoM + tipoB + tipoC + tipoD, datos.dummy)
print(m2)
```

Finalmente, para usar la variable categórica como predictor, agregamos al modelo todas las variables artificiales creadas a partir de ella. Cabe señalar que la función `lm()` realiza internamente este proceso cuando recibe una variable categórica entre los predictores (las variables indicadoras descartadas en el script 15.2 replican el resultado que entrega `lm()`). No obstante, al usar el modelo, debemos fijarnos en que la cantidad de predictores categóricos sea la misma, como también en que la cantidad y el orden de sus niveles coincida con los del conjunto de entrenamiento.

El script 15.2 ajusta además dos modelos, el primero usando el conjunto de datos original y el segundo, el conjunto de datos transformado para que tenga variables indicadoras en reemplazo de las variables categóricas. Si nos fijamos en la figura 15.3, podemos ver que en ambos casos el modelo tiene los mismos predictores.

```
Call:
lm(formula = valor ~ sexo + tipo, data = datos)

Coefficients:
(Intercept)        sexoM        tipoB        tipoC        tipoD
     1.2139       0.8191       0.3465       0.5970       1.4213


Call:
lm(formula = valor ~ sexoM + tipoB + tipoC + tipoD, data = datos.dummy)

Coefficients:
(Intercept)        sexoM        tipoB        tipoC        tipoD
     1.2139       0.8191       0.3465       0.5970       1.4213
```

Figura 15.3: resultado del script 15.2.

15.2 CONDICIONES PARA USAR RLM

Llegado este punto, necesitamos examinar con más detalle las condiciones que debemos cumplir para que un modelo de regresión lineal sea generalizable:

1. Las variables predictoras deben ser cuantitativas o dicotómicas (de ahí la necesidad de variables indicadoras para manejar más de dos niveles).
2. La variable de respuesta debe ser cuantitativa y continua, sin restricciones para su variabilidad.
3. Los predictores deben tener algún grado de variabilidad (su varianza no debe ser igual a cero). En otras palabras, no pueden ser constantes.
4. No debe existir **multicolinealidad**. Esto significa que no deben existir relaciones lineales *fuertes* entre dos o más predictores (coeficientes de correlación altos).
5. Los residuos deben ser homocedásticos (con varianzas similares) para cada nivel de los predictores.
6. Los residuos deben seguir una distribución cercana a la normal centrada en cero.
7. Los valores de la variable de respuesta son independientes entre sí.
8. Cada predictor se relaciona linealmente con la variable de respuesta.

15.3 EVALUACIÓN DEL AJUSTE DE UNA RLM

En el capítulo anterior introdujimos el coeficiente de determinación (R^2) como instrumento para evaluar la bondad de ajuste de una regresión lineal. Sin embargo, cuando el modelo es multivariado, la función ya conocida para estimar R^2 genera una mala estimación del porcentaje de la varianza explicada por el modelo, pues los grados de libertad asociados a la variabilidad de los residuos es ahora diferente, como muestra la ecuación 15.3, donde n es el tamaño de la muestra y k, la cantidad de variables predictoras.

$$\nu = n - k - 1 \tag{15.3}$$

Así, para evaluar una RLM tenemos que usar un coeficiente de determinación ajustado. Algunos autores han propuesto distintas maneras de efectuar este ajuste, una de las cuales presentamos en la ecuación 15.4. También podemos usar este ajuste cuando tenemos un único predictor, aunque la diferencia en este caso suele ser muy pequeña como para ser relevante.

$$R^2 = 1 - \frac{s_e^2}{s_y^2} \cdot \frac{n-1}{n-k-1} \tag{15.4}$$

Existen otras alternativas para evaluar la bondad de ajuste de un modelo que se basan en el **principio de parsimonia**, también llamado navaja de Occam, el cual indica que un modelo debe mantenerse tan simple como sea posible. Dos de ellas son el **criterio de información de Akaike**, abreviado AIC, y el **criterio bayesiano de Schwarz** (BIC o SBC), que penalizan el modelo por contener variables adicionales, por lo que mientras menor sea su valor, mejor será el modelo. Si bien el cálculo de estas medidas no se detalla aquí por ser un tópico más avanzado, podemos obtenerlas en R mediante las funciones `AIC(object)` y `BIC(object)`, donde `object` corresponde a un modelo lineal ajustado.

Para el modelo que habíamos ajustado usando únicamente el peso como predictor, obtenemos $AIC = 166,03$ y $BIC = 170,43$. Del mismo modo, para el modelo que usa como predictores el peso y el cuarto de milla, en cambio, tenemos que que $AIC = 156,72$ y $BIC = 162,58$. En consecuencia, el segundo modelo parece ser "mejor" bajo estos criterios.

Otra opción, adecuada cuando necesitamos saber cuáles predictores son estadísticamente significativos, es observar los valores p asociados a cada predictor. Habitualmente consideraremos significativos aquellos predictores para los cuales $p < 0,05$.

15.4 COMPARACIÓN DE MODELOS

En la sección anterior vimos que métricas como el AIC o el BIC nos pueden resultar útiles para comparar dos modelos de regresión lineal, considerando la noción general que un modelo es mejor mientras menor sea su valor de AIC (o BIC). Si calculamos el AIC para cada uno de los modelos ajustados hasta ahora (script 15.3), veremos que el AIC del modelo con dos predictores es menor. Sin embargo, al ser una medida relativa, hasta ahora no contamos con una prueba estadística que nos permita determinar si la diferencia es significativa.

Cuando los modelos son jerárquicos, es decir, el segundo incorpora nuevos predictores además de mantener los del primer modelo, podemos hacer una prueba de hipótesis usando los coeficientes de determinación para ver si la diferencia es significativa. El estadístico de prueba se calcula mediante la ecuación 15.5, donde:

- n: cantidad de observaciones.
- k_2: cantidad de predictores en el modelo con más variables (segundo modelo).
- R^2_{cambio}: diferencia entre los valores de R^2 del segundo y el primer modelo.
- k_{cambio}: diferencia entre la cantidad de predictores del segundo y el primer modelo.
- R_2^2: coeficiente de determinación del segundo modelo.

$$F_{cambio} = \frac{(n-k_2-1)R^2_{cambio}}{k_{cambio}(1-R_2^2)} \tag{15.5}$$

Así, para los dos modelos ajustados hasta ahora tenemos:

$$F_{cambio} = \frac{(32-2-1)(0,8264-0,7528)}{1(1-0,8264)} = 12,295$$

Notemos que el estadístico de prueba sigue una distribución F con k_{cambio} y $n-k_2-1$ grados de libertad, a partir de lo cual podemos determinar el valor p correspondiente mediante la llamada `pf(12.295, 1, 29, lower.tail = FALSE)`, obteniendo como resultado $p = 0,0015$. En consecuencia, podemos concluir con 95 % de confianza que la diferencia es significativa, por lo que el segundo modelo es mejor.

Como ya es habitual, en R podemos hacer esta tarea de forma simple gracias a la función `anova(object, ...)`, que recibe como argumentos los diferentes modelos a comparar. La interpretación del resultado de esta prueba es sencilla: si el valor p obtenido es significativo, entonces el modelo más complejo (con más predictores) es mejor. Al ejecutar el script 15.3, podemos ver que obtenemos el mismo resultado que con la prueba F.

Script 15.3: comparación de dos modelos lineales.

```
# Cargar datos.
datos <- mtcars

# Ajustar modelo con el peso como predictor.
modelo_1 <- lm(mpg ~ wt, data = datos)
print(summary(modelo_1))
aic_1 <- AIC(modelo_1)
cat("Modelo 1: AIC =", AIC(modelo_1), "\n")

# Ajustar modelo con el peso y el cuarto de milla como predictores.
modelo_2 <- lm(mpg ~ wt + qsec, data = datos)
print(summary(modelo_2))
aic_2 <- AIC(modelo_2)
cat("Modelo 2: AIC =", AIC(modelo_2), "\n")

# Comparar ambos modelos.
comparacion <- anova(modelo_1, modelo_2)
print(comparacion)
```

15.5 SELECCIÓN DE PREDICTORES

La cuarta condición para emplear RLM indica que debemos evitar la multicolinealidad. Esto es importante porque el ajuste de un modelo RLM asume que podemos cambiar una variable predictora, *manteniendo las otras constantes*. Cuando las variables predictoras están correlacionadas, se hace imposible cambiar el valor de una sin alterar también a las demás, desestabilizando la estimación de los coeficientes del modelo que indican cómo influye cada variable predictora en la variable de salida de forma *independiente*.

Cuando existe colinealidad, los valores de los coeficientes varían enormemente si se agregan o quitan unos pocos datos de entrenamiento y se reduce el poder estadístico del modelo. Por esta razón, es importante que escojamos con cuidado las variables predictoras a considerar en un modelo RML. Es más, existe un riesgo adicional cuando los predictores están correlacionados: el orden en que se agreguen puede influenciar la calidad del modelo. En consecuencia, necesitamos contar con alguna estrategia que nos permita determinar cuáles predictores incluir. Existen diversas alternativas para esta tarea.

El método más adecuado, aunque también el más complejo, es la **regresión jerárquica**. Es el que debemos considerar al momento de intentar probar una teoría y consiste en comenzar por incorporar en primer lugar aquellos predictores ya conocidos, en orden de importancia, en base a investiagiones previas. Una vez incorporados todos los predictores ya conocidos, podemos incorporar otros nuevos si creemos que existen **buenas y justificadas razones** para ello. Antes de la masificación de los computadores y de entornos como R, ¡esta era la única alternativa viable!

En R, podemos realizar este método con ayuda de la función `update(object, formula)`, que nos permite incorporar o quitar variables del modelo, donde:

- `object`: modelo previamente ajustado, en este caso con `lm()`.
- `formula`: actualización de la fórmula para el nuevo modelo.

En este caso no contamos con investigaciones previas que nos permitan formular una teoría, por lo que nos limitaremos a proporcionar un ejemplo de cómo usar esta función (script 15.4), cuyos resultados presentamos en la figura 15.4.

Script 15.4: incorporación y eliminación de variables en un modelo de RLM.

```r
# Cargar datos.
datos <- mtcars

# Ajustar modelo inicial con la variable wt como predictor.
modelo <- lm(mpg ~ wt, data = datos)
cat("=== Modelo inicial ===\n")
print(modelo)

# Incorporar el predictor cyl.
modelo <- update(modelo, . ~ . + cyl)
cat("=== Modelo con predictores wt y cyl ===\n")
print(modelo)

# Quitar el predictor wt.
modelo <- update(modelo, . ~ . - wt)
cat("=== Modelo con predictor cyl ===\n")
print(modelo)

# Agregar predictores wt y drat, y quitar predictor cyl.
modelo <- update(modelo, . ~ . + wt + drat - cyl)
cat("=== Modelo con predictores wt y drat ===\n")
print(modelo)
```

Si, en lugar de probar una teoría, lo que queremos es **explorar los datos**, podemos usar otras estrategias.

Selección hacia adelante Se crea un modelo inicial nulo, es decir, sin predictores, para el cual únicamente se estima la intercepción. A continuación, se escoge como primer predictor aquel que tenga la correlación más alta con la variable de respuesta. Si dicho predictor incrementa la capacidad predictiva del modelo, se retiene en el modelo y se procede a seleccionar un segundo predictor. El modelo con una única variable ajustado al inicio de este capítulo tiene como predictor el peso de los automóviles, variable que en efecto tiene la más alta correlación con el rendimiento (variable de respuesta). El coeficiente de determinación para este modelo es $R^2 = 0,7528$, lo cual significa que explica aproximadamente el 75 % de la variabilidad de la respuesta. En consecuencia, existe aún un 25 % de variabilidad de la respuesta que aún no ha sido explicado.
Para la selección de predictores adicionales, en cada repetición se escoge aquel predictor (que no haya sido previamente agregado al modelo) que tenga la **máxima correlación semi-parcial con la respuesta**, es decir, que explique la máxima porción de la varianza no cubierta por el modelo ya existente. Si la inclusión de este nuevo predictor mejora el poder predictivo del modelo, se incorpora de manera definitiva y se evalúa el siguiente predictor.
Adicionalmente, se evalúa si la inclusión de cada nuevo predictor mejora (es decir, reduce) el AIC. Si ninguno de los posibles predictores restantes logra reducir este indicador, se detiene la inclusión de nuevos predictores.

Eliminación hacia atrás Es el proceso inverso a la selección hacia adelante, puesto que se comienza desde un modelo con todas las variables para luego eliminar predictores uno a uno y evaluar el AIC. Si este último se reduce, se elimina dicho predictor y se reevalúa la contribución de los predictores que aún se encuentran en el modelo. Una vez más, el proceso se repite hasta que no es posible reducir el AIC.

Regresión escalonada En términos generales, opera de manera análoga a la selección hacia adelante. Sin embargo, para reducir el potencial efecto debido al orden de incorporación de los predictores, cada vez que se incorpora uno nuevo se evalúa qué ocurre al descartar el menos importante. En consecuencia, el modelo es permanentemente reevaluado para descartar posibles redundancias entre predictores.

```
=== Modelo inicial ===

Call:
lm(formula = mpg ~ wt, data = datos)

Coefficients:
(Intercept)           wt
     37.285       -5.344

=== Modelo con predictores wt y cyl ===

Call:
lm(formula = mpg ~ wt + cyl, data = datos)

Coefficients:
(Intercept)           wt          cyl
     39.686       -3.191       -1.508

=== Modelo con predictor cyl ===

Call:
lm(formula = mpg ~ cyl, data = datos)

Coefficients:
(Intercept)          cyl
     37.885       -2.876

=== Modelo con predictores wt y drat ===

Call:
lm(formula = mpg ~ wt + drat, data = datos)

Coefficients:
(Intercept)           wt         drat
     30.290       -4.783        1.442
```

Figura 15.4: resultado del script 15.4.

Todos los subconjuntos Una alternativa más exhaustiva, altamente costosa en tiempo de ejecución, es usar un algoritmo de fuerza bruta en el que se exploran todos los subconjuntos de predictores.

Es importante reiterar que solo debemos usar estos métodos si estamos **explorando datos**, pues de lo contrario incurriríamos en faltas a la ética. Veamos por ejemplo el trabajo de Montero Muñoz y col. (2012), donde estudian algunas implicaciones clínicas en el uso de antibióticos en ancianos mayores de 80 años. Podemos ver que, en la recolección de datos, recopilaron variables que la medicina ha probado a lo largo de su historia que son importantes al evaluar la salud de un paciente y que pueden ser afectadas por el uso de medicamentos. Un equipo poco ético podría haber empleado otras variables registradas en las bases de datos de origen, por ejemplo, el monto de la pensión del paciente. ¿Qué consecuencias podrían existir si tuviéramos una correlación espuria de esta variable con la variable de respuesta? ¡Podríamos afectar las vidas de millones de personas si tal estudio concluyera que los antibióticos son más nocivos para ancianos con bajas pensiones!

Todas las estrategias mencionadas están disponibles en R con ayuda de la ya conocida función `update()` y las funciones `add1(object, scope)` y `drop1(object, scope)`, donde:

- `object`: un modelo ajustado.
- `scope`: fórmula que proporciona los términos a agregar o quitar.

La función `add1()` evalúa la incorporación de cada nuevo predictor potencial (separadamente) a un modelo base y entrega algunas métricas para el efecto que tiene su incorporación, entre ellas el AIC. El mejor nuevo predictor corresponde, entonces, a aquella variable con el menor AIC. Las líneas 15 y 22 del script 15.5 ilustran su uso, obteniéndose los resultados que se muestran en la figura 15.5.

```
Single term additions

Model:
mpg ~ 1
       Df Sum of Sq     RSS     AIC
<none>              1126.05 115.943
cyl     1    817.71  308.33  76.494
disp    1    808.89  317.16  77.397
hp      1    678.37  447.67  88.427
drat    1    522.48  603.57  97.988
wt      1    847.73  278.32  73.217
qsec    1    197.39  928.66 111.776
vs      1    496.53  629.52  99.335
am      1    405.15  720.90 103.672
gear    1    259.75  866.30 109.552
carb    1    341.78  784.27 106.369


Single term additions

Model:
mpg ~ wt
       Df Sum of Sq    RSS    AIC
<none>              278.32 73.217
cyl     1    87.150 191.17 63.198
disp    1    31.639 246.68 71.356
hp      1    83.274 195.05 63.840
drat    1     9.081 269.24 74.156
qsec    1    82.858 195.46 63.908
vs      1    54.228 224.09 68.283
am      1     0.002 278.32 75.217
gear    1     1.137 277.19 75.086
carb    1    44.602 233.72 69.628
```

Figura 15.5: resultado de las llamadas a `add1()` en el script 15.5

De manera similar, la función `drop1()` evalúa (separadamente) la eliminación potencial de cada predictor presente en un modelo base y entrega las mismas métricas que `add1()` para el efecto que tiene su eliminación. El mejor predictor a descartar es, una vez más, aquel que lleva a la mayor reducción en AIC. La línea 29 del script 15.5 ilustra su uso, obteniéndose los resultados que se muestran en la figura 15.6.

Script 15.5: Evaluación de variables a incorporar y eliminar en un modelo de RLM.

```
1 # Cargar datos.
2 datos <- mtcars
3
4 # Ajustar modelo nulo.
5 nulo <- lm(mpg ~ 1, data = datos)
```

```
Single term deletions

Model:
mpg ~ cyl + disp + hp + drat + wt + qsec + vs + am + gear + carb
       Df Sum of Sq    RSS    AIC
<none>              147.49 70.898
cyl     1    0.0799 147.57 68.915
disp    1    3.9167 151.41 69.736
hp      1    6.8399 154.33 70.348
drat    1    1.6270 149.12 69.249
wt      1   27.0144 174.51 74.280
qsec    1    8.8641 156.36 70.765
vs      1    0.1601 147.66 68.932
am      1   10.5467 158.04 71.108
gear    1    1.3531 148.85 69.190
carb    1    0.4067 147.90 68.986
```

Figura 15.6: resultado de la llamada a `drop1()` en el script 15.5

```r
# cat("=== Modelo nulo ===\n")
# print(summary(nulo))

# Ajustar modelo completo.
completo <- lm(mpg ~ ., data = datos)
# cat("=== Modelo completo ===\n")
# print(summary(completo))

# Evaluar variables para incorporar.
print(add1(nulo, scope = completo))
cat("\n\n")

# Agregar la variable con menor AIC.
modelo <- update(nulo, . ~ . + wt)

# Evaluar variables para incorporar.
print(add1(modelo, scope = completo))
cat("\n\n")

# Agregar la variable con menor AIC.
modelo <- update(modelo, . ~ . + cyl)

# Evaluar variables para eliminar.
print(drop1(completo, scope = completo))
cat("\n\n")

# Eliminar la variable con menor AIC.
modelo <- update(modelo, . ~ . - cyl)
```

Por supuesto, R en la práctica ya cuenta con funciones que implementan los métodos para seleccionar predictores antes descritos (excepto la regresión jerárquica, por supuesto). Los tres primeros pueden efectuarse mediante la función `step(object, scope, direction, trace)`, que usa `add1()` y `drop1()` de manera iterativa, donde:

- `object`: es un modelo ya ajustado que es usado como punto de partida.
- `scope`: es una lista de fórmulas que define el rango de modelos a explorar.
- `direction`: indica el tipo de selección a realizar, donde "`forward`" corresponde a selección hacia ade-

lante; "`backward`", a eliminación hacia atrás, y "`both`", a regresión escalonada.
- `trace`: argumento opcional que indica si se quiere ver por consola el proceso realizado.

El script 15.6 muestra el funcionamiento de la función `step()` para seleccionar los predictores a incorporar en un modelo donde la respuesta es, una vez más, el rendimiento de un automóvil.

Script 15.6: selección de predictores a incluir en una RLM.

```
1  library(leaps)
2
3  # Cargar datos.
4  datos <- mtcars
5
6  # Ajustar modelo nulo.
7  nulo <- lm(mpg ~ 1, data = datos)
8  cat("=== Modelo nulo ===\n")
9  print(summary(nulo))
10
11 # Ajustar modelo completo.
12 completo <- lm(mpg ~ ., data = datos)
13 cat("=== Modelo completo ===\n")
14 print(summary(completo))
15
16 # Ajustar modelo con selección hacia adelante.
17 adelante <- step(nulo, scope = list(upper = completo), direction = "forward",
18                  trace = 0)
19
20 cat("=== Modelo con selección hacia adelante ===\n")
21 print(summary(adelante))
22 cat("AIC =", AIC(adelante), "\n\n")
23
24 # Ajustar modelo con eliminación hacia atrás.
25 atras <- step(completo, scope = list(lower = nulo), direction = "backward",
26               trace = 0)
27
28 cat("=== Modelo con eliminación hacia atrás ===\n")
29 print(summary(atras))
30 cat("AIC =", AIC(atras), "\n\n")
31
32 # Ajustar modelo con regresión escalonada.
33 escalonado <- step(nulo, scope = list(lower = nulo, upper = completo),
34                    direction = "both", trace = 0)
35
36 cat("=== Modelo con regresión escalonada ===\n")
37 print(summary(escalonado))
38 cat("AIC =", AIC(escalonado), "\n\n")
39
40 # Ajustar modelo con todos los subconjuntos.
41 modelos <- regsubsets(mpg ~ ., data = datos, method = "exhaustive",
42                       nbest = 1, nvmax = 10)
43
44 print(plot(modelos))
```

La línea 7 del script 15.6 ajusta el modelo nulo, obteniéndose el resultado que se muestra en la figura 15.7. De manera similar, la línea 12 ajusta el modelo completo (figura 15.8).

Las líneas 17–18 usan el método de selección hacia adelante, comenzando desde el modelo nulo. Notemos que en el argumento `scope` entregamos, en este caso, el modelo completo con todos los posibles predictores. No obstante, no es necesario siempre comenzar desde el modelo nulo o evaluar hasta el modelo completo.

```
Call:
lm(formula = mpg ~ 1, data = datos)

Residuals:
    Min      1Q  Median      3Q     Max
-9.6906 -4.6656 -0.8906  2.7094 13.8094

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   20.091      1.065   18.86   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6.027 on 31 degrees of freedom
```

Figura 15.7: modelo nulo.

En la llamada a `step()` de las líneas 17–18, el argumento `trace` tiene por defecto el valor 1, por lo que la función muestra información de los diferentes modelos evaluados (las figuras 15.9 y 15.10 muestran esta información para todas las iteraciones). Notemos que se comienza por calcular el AIC para el modelo nulo, y luego se determina el AIC (y otras métricas) que se obtendría para diferentes modelos, cada uno de los cuales contendría solo una de las variables restantes. El predictor que se escoge es aquel que genera el modelo con menor AIC (el peso del automóvil en la iteración 1).

En la siguiente iteración se conserva el peso como predictor y se evalúa el AIC de los modelos que incorporan un predictor adicional. Una vez más, se incorpora aquel con menor AIC, correspondiente en esta ocasión a la cantidad de cilindros.

El proceso se detiene una vez que ninguno de los predictores logra un AIC menor que el del modelo ya obtenido en la iteración previa. El modelo final obtenido mediante selección hacia adelante se presenta en la figura 15.11, que se consigue en tres iteraciones y considera, además del peso del automóvil y el número de cilindros, los caballos de fuerza bruta (`hp`).

De manera similar, las figuras 15.12 y 15.13 muestran los modelos resultantes al usar eliminación hacia atrás y regresión escalonada, respectivamente (líneas 25–34). Notemos que, en estos casos, la llamada a `step()` se realiza con el argumento `trace = 0`, por lo que no nos muestra información acerca de los diferentes modelos evaluados.

Las líneas 41–42 del script 15.6, por otra parte, muestran la aplicación del método de todos los subconjuntos para determinar el mejor modelo, proceso que hacemos mediante la función `regsubsets(formula, data, nbest, nvmax, force.in, force.out, method = ‘‘exhaustive’’)` del paquete `leaps`, donde:

- `formula`: indica la variable de respuesta y los posibles predictores.
- `data`: matriz de datos.
- `nbest`: cantidad de modelos a reportar por cada tamaño de subconjunto. Si, por ejemplo, `nbest = 3`, entonces se reportan los tres mejores modelos con un único predictor, los tres mejores modelos con dos predictores, etc.
- `nvmax`: fija una cantidad máxima de predictores a considerar.
- `force.in`: argumento opcional que toma la forma de un vector con los índices de las columnas que deben ser forzosamente consideradas en los modelos evaluados.
- `force.out`: argumento opcional que toma la forma de un vector con los índices de las columnas que deben ser forzosamente excluidas en los modelos evaluados.

La figura 15.14 representa gráficamente el resultado de la aplicación del método de todos los subconjuntos. En ella, el eje x lista todas las variables predictoras y el eje y corresponde al BIC de cada modelo, que es el criterio utilizado por defecto por la función `regsubsets()`. Fijémonos en que el eje y no es una escala graduada, sino

```
Call:
lm(formula = mpg ~ ., data = datos)

Residuals:
    Min      1Q  Median      3Q     Max
-3.4506 -1.6044 -0.1196  1.2193  4.6271

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 12.30337   18.71788   0.657   0.5181
cyl         -0.11144    1.04502  -0.107   0.9161
disp         0.01334    0.01786   0.747   0.4635
hp          -0.02148    0.02177  -0.987   0.3350
drat         0.78711    1.63537   0.481   0.6353
wt          -3.71530    1.89441  -1.961   0.0633 .
qsec         0.82104    0.73084   1.123   0.2739
vs           0.31776    2.10451   0.151   0.8814
am           2.52023    2.05665   1.225   0.2340
gear         0.65541    1.49326   0.439   0.6652
carb        -0.19942    0.82875  -0.241   0.8122
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.65 on 21 degrees of freedom
Multiple R-squared:  0.869,Adjusted R-squared:  0.8066
F-statistic: 13.93 on 10 and 21 DF,  p-value: 3.793e-07
```

Figura 15.8: modelo completo.

una lista ordenada del BIC para cada uno de los modelos evaluados. Así, la figura corresponde a una matriz en que cada fila está asignada a un modelo, donde se colorea el recuadro para las variables presentes en dicho modelo. Así, el mejor modelo (es decir, el que tiene el menor BIC) es aquel que contiene como predictores el peso (`wt`), el cuarto de milla (`qsec`) y el tipo de transmisión (`am`), además de la intercepción.

También es importante fijarnos en que, en este caso particular, obtenemos el mismo modelo al usar selección hacia adelante y regresión escalonada. Del mismo modo, la eliminación hacia atrás y el método de todos los subconjuntos generan un mismo modelo. Si usamos el AIC como indicador, el primero de estos dos modelos es el mejor. No obstante, el segundo es mejor si consideramos el BIC.

15.6 EVALUACIÓN DE UN MODELO DE RLM

Para ilustrar el proceso de evaluación de un modelo de RLM, consideremos el modelo obtenido mediante eliminación hacia atrás en la sección precedente, es decir, el modelo que tiene como predictores el peso, el cuarto de milla y el tipo de marcha. Al graficar este modelo (script 15.7, línea 6) obtenemos los gráficos de la figura 15.15, donde podemos ver que la distribución de los residuos se desvía un poco de la normal y que parece haber algunas observaciones atípicas influenciando el ajuste del modelo.

```
Start:  AIC=115.94
mpg ~ 1

       Df Sum of Sq     RSS     AIC
+ wt    1    847.73  278.32  73.217
+ cyl   1    817.71  308.33  76.494
+ disp  1    808.89  317.16  77.397
+ hp    1    678.37  447.67  88.427
+ drat  1    522.48  603.57  97.988
+ vs    1    496.53  629.52  99.335
+ am    1    405.15  720.90 103.672
+ carb  1    341.78  784.27 106.369
+ gear  1    259.75  866.30 109.552
+ qsec  1    197.39  928.66 111.776
<none>               1126.05 115.943

Step:  AIC=73.22
mpg ~ wt

       Df Sum of Sq    RSS    AIC
+ cyl   1    87.150 191.17 63.198
+ hp    1    83.274 195.05 63.840
+ qsec  1    82.858 195.46 63.908
+ vs    1    54.228 224.09 68.283
+ carb  1    44.602 233.72 69.628
+ disp  1    31.639 246.68 71.356
<none>               278.32 73.217
+ drat  1     9.081 269.24 74.156
+ gear  1     1.137 277.19 75.086
+ am    1     0.002 278.32 75.217
```

Figura 15.9: modelos evaluados por la función `step()` durante el proceso de selección hacia adelante (parte 1).

15.6.1 Identificación de valores con sobreinfluencia

Existen diversos estadísticos que nos permiten evaluar la influencia de una observación en el ajuste de un modelo de regresión lineal:

1. **Residuo estandarizado**: los residuos deben seguir una distribución normal estándar, por lo que se esperaría que el 95 % de ellos se encuentre entre -1,96 y 1,96, y el 99 % entre -2,58 y 2,58.
2. **Valor predicho ajustado:** corresponde al valor predicho si se excluyera dicho punto en el ajuste del modelo. Si el punto no ejerce gran influencia en el ajuste del modelo, se esperaría que este valor fuera muy cercano al predicho cuando dicho punto sí es considerado para el ajuste.
3. **Residuo estudiantizado:** está dado por el valor predicho ajustado dividido por el error estándar. Una característica importante de esta medida es que es estandarizada y sigue una distribución t, por lo que puede emplearse para hacer comparaciones entre distintos modelos de regresión. Sin embargo, esta medida solo indica cuánto influye la presencia de un punto en el conjunto de entrenamiento en su valor predicho, pero no proporciona información alguna en cuanto a la influencia de la observación en el modelo como un todo.
4. **Diferencia en ajuste:** más conocido como DFFit, es la diferencia entre el valor predicho para la observación evaluada cuando esta es considerada en el ajuste del modelo y cuando no lo es.
5. **Diferencia en betas:** más conocido como DFBeta, corresponde a la diferencia entre los valores de un

```
Step:  AIC=63.2
mpg ~ wt + cyl

       Df Sum of Sq    RSS    AIC
+ hp    1   14.5514 176.62 62.665
+ carb  1   13.7724 177.40 62.805
<none>              191.17 63.198
+ qsec  1   10.5674 180.60 63.378
+ gear  1    3.0281 188.14 64.687
+ disp  1    2.6796 188.49 64.746
+ vs    1    0.7059 190.47 65.080
+ am    1    0.1249 191.05 65.177
+ drat  1    0.0010 191.17 65.198

Step:  AIC=62.66
mpg ~ wt + cyl + hp

       Df Sum of Sq    RSS    AIC
<none>              176.62 62.665
+ am    1    6.6228 170.00 63.442
+ disp  1    6.1762 170.44 63.526
+ carb  1    2.5187 174.10 64.205
+ drat  1    2.2453 174.38 64.255
+ qsec  1    1.4010 175.22 64.410
+ gear  1    0.8558 175.76 64.509
+ vs    1    0.0599 176.56 64.654
```

Figura 15.10: modelos evaluados por la función `step()` durante el proceso de selección hacia adelante (parte 2).

parámetro cuando es estimado usando todas las observaciones y cuando es estimado sin considerar la observación evaluada. Se calcula para cada parámetro del modelo. Se consideran preocupantes aquellas observaciones en que este estimador es mayor a 1.

6. **Distancia de Cook:** es una medida del efecto que tiene una observación en particular combinadamente en todos los parámetros de un modelo. Aquellos valores para los cuales la distancia de Cook sea mayor a 1 pueden ser considerados como potencialmente problemáticos.
7. **Apalancamiento:** estima la influencia del valor observado en los valores predichos. Toma valores entre 0 y 1. Un apalancamiento igual a 0 señala que un punto no ejerce influencia alguna, mientras que un valor de 1 indica que la influencia ejercida por esa observación es total. Se consideran preocupantes aquellas observaciones para las cuales esta medida supere en dos o tres veces el apalancamiento promedio, dado por la ecuación 15.6, donde k es la cantidad de predictores en el modelo y n la cantidad de observaciones empleadas para el ajuste.

$$\overline{x}_{Apalancamiento} = \frac{k+1}{n} \tag{15.6}$$

8. **Razón de covarianza:** corresponde a la razón entre los determinantes de la matriz de covarianzas cuando se consideran todas las observaciones y cuando se omite la observación en estudio. Aquellas observaciones para las cuales el valor de esta medida estén fuera del intervalo definido por la ecuación 15.7 se consideran preocupantes.

$$|Covratio - 1| < \frac{3(k+1)}{n} \tag{15.7}$$

```
Call:
lm(formula = mpg ~ wt + cyl + hp, data = datos)

Residuals:
    Min      1Q  Median      3Q     Max
-3.9290 -1.5598 -0.5311  1.1850  5.8986

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 38.75179    1.78686  21.687  < 2e-16 ***
wt          -3.16697    0.74058  -4.276 0.000199 ***
cyl         -0.94162    0.55092  -1.709 0.098480 .
hp          -0.01804    0.01188  -1.519 0.140015
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.512 on 28 degrees of freedom
Multiple R-squared:  0.8431,Adjusted R-squared:  0.8263
F-statistic: 50.17 on 3 and 28 DF,  p-value: 2.184e-11

AIC = 155.4766
```

Figura 15.11: modelo obtenido mediante selección hacia adelante.

El script 15.7 muestra la detección de posibles valores atípicos que puedan estar influenciando el modelo de la figura 15.12, obteniéndose los resultados presentados en la figura 15.16. Al examinar estos resultados, en conjunto con los gráficos de la figura 15.15, las observaciones correspondientes al Chrysler Imperial, el Fiat 128 y el Toyota Corolla parecen ser candidatos a eliminación para la generación del modelo. Sin embargo, la distancia de Cook estimada para todas las observaciones potencialmente influyentes están lejos de sobrepasar el valor recomendado, por lo que en este caso no parece ser necesario quitarlos, aun cuando algunas muestren valores un tanto alto de apalancamiento y covarianza. Eliminar datos para construir un modelo debe tener una **sólida justificación**, y por ningún motivo debe hacerse por conveniencia, lo que sería **profundamente antiético**. Consideremos, por ejemplo, que la *pobreza extrema* o los *super-ricos* son casos extremos que, usualmente, no deben eliminarse de un modelo que intente estudiar la distribución de los ingresos de un país.

Script 15.7: identificación de valores atípicos.

```r
# Cargar datos.
datos <- mtcars

# Ajustar modelo.
modelo <- lm(mpg ~ wt + qsec + am, data = datos)
plot(modelo)

# Reducir matriz de datos para que solo contenga los predictores
# empleados y la respuesta.
predictores <- names(coef(modelo))[-1]
datos <- datos[, c(predictores, "mpg")]

# Construir una matriz de datos con la respuesta predicha, los
# residuos y algunas estadísticas para evaluar la influencia de
# cada observación.
resultados <- data.frame(respuesta_predicha = fitted(modelo))
```

```
Call:
lm(formula = mpg ~ wt + qsec + am, data = datos)

Residuals:
    Min      1Q  Median      3Q     Max
-3.4811 -1.5555 -0.7257  1.4110  4.6610

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   9.6178     6.9596   1.382 0.177915
wt           -3.9165     0.7112  -5.507 6.95e-06 ***
qsec          1.2259     0.2887   4.247 0.000216 ***
am            2.9358     1.4109   2.081 0.046716 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.459 on 28 degrees of freedom
Multiple R-squared:  0.8497,Adjusted R-squared:  0.8336
F-statistic: 52.75 on 3 and 28 DF,  p-value: 1.21e-11

AIC = 154.1194
```

Figura 15.12: modelo obtenido mediante eliminación hacia atrás.

```
Call:
lm(formula = mpg ~ wt + cyl + hp, data = datos)

Residuals:
    Min      1Q  Median      3Q     Max
-3.9290 -1.5598 -0.5311  1.1850  5.8986

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 38.75179    1.78686  21.687  < 2e-16 ***
wt          -3.16697    0.74058  -4.276 0.000199 ***
cyl         -0.94162    0.55092  -1.709 0.098480 .
hp          -0.01804    0.01188  -1.519 0.140015
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.512 on 28 degrees of freedom
Multiple R-squared:  0.8431,Adjusted R-squared:  0.8263
F-statistic: 50.17 on 3 and 28 DF,  p-value: 2.184e-11

AIC = 155.4766
```

Figura 15.13: modelo obtenido mediante regresión escalonada.

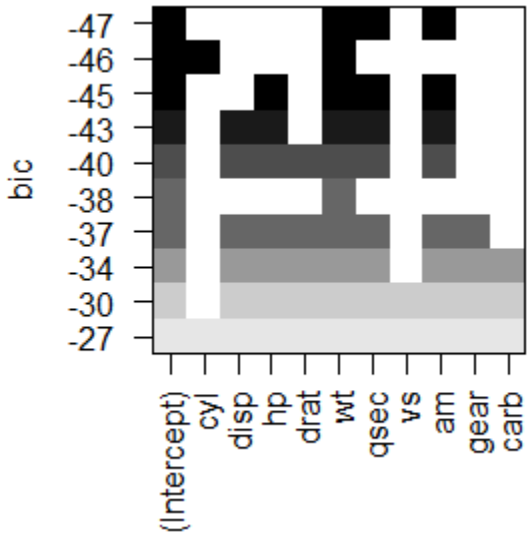
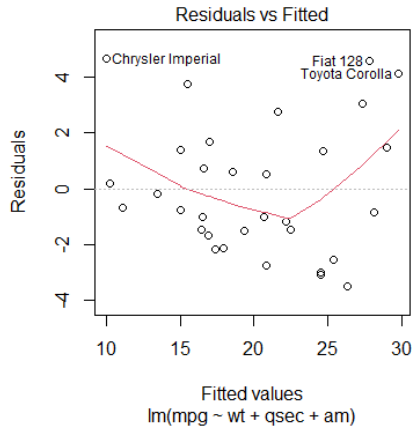
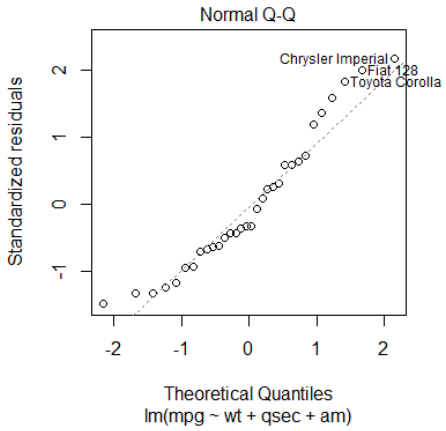


Figura 15.14: representación gráfica de los mejores modelos encontrados mediante el método de todos los subconjuntos.

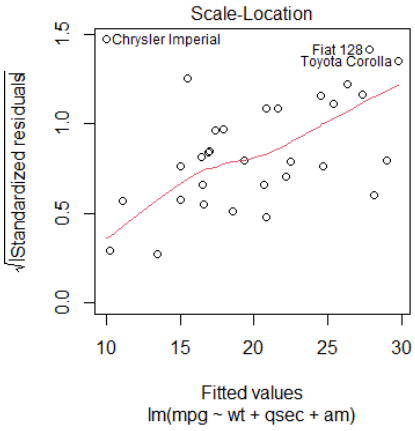
```
resultados[["residuos_estandarizados"]] <- rstandard(modelo)
resultados[["residuos_estudiantizados"]] <-rstudent(modelo)
resultados[["distancia_Cook"]] <- cooks.distance(modelo)
resultados[["dfbeta"]] <- dfbeta(modelo)
resultados[["dffit"]] <- dffits(modelo)
resultados[["apalancamiento"]] <- hatvalues(modelo)
resultados[["covratio"]] <- covratio(modelo)

cat("Identificación de valores atípicos:\n")
# Observaciones con residuos estandarizados fuera del 95% esperado.
sospechosos1 <- which(abs(
  resultados[["residuos_estandarizados"]]) > 1.96)

cat("- Residuos estandarizados fuera del 95% esperado:",
    sospechosos1, "\n")

# Observaciones con distancia de Cook mayor a uno.
sospechosos2 <- which(resultados[["cooks.distance"]] > 1)

cat("- Residuos con una distancia de Cook alta:",
    sospechosos2, "\n")

# Observaciones con apalancamiento mayor igual al doble del
# apalancamiento promedio.

apal_medio <- (ncol(datos) + 1) / nrow(datos)
sospechosos3 <- which(resultados[["apalancamiento"]] > 2 * apal_medio)

cat("- Residuos con apalancamiento fuera de rango:",
    sospechosos3, "\n")

# Observaciones con DFBeta mayor o igual a 1.
sospechosos4 <- which(apply(resultados[["dfbeta"]] >= 1, 1, any))
names(sospechosos4) <- NULL

cat("- Residuos con DFBeta >= 1:",
```



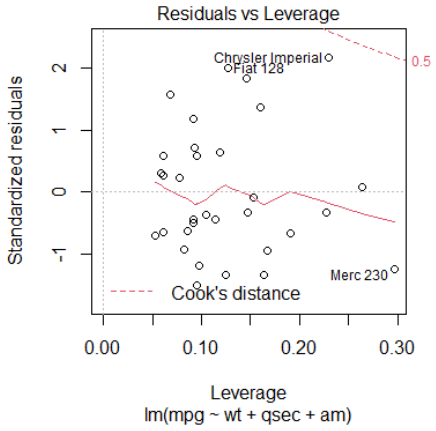
(a) residuos.



(b) distribución de los residuos.



(c) residuos estandarizados.



(d) apalancamiento.

Figura 15.15: gráficos disponibles en R (base) para evaluar un modelo lineal.

```
    sospechosos4, "\n")

# Observaciones con razón de covarianza fuera de rango.
inferior <- 1 - 3 * apal_medio
superior <- 1 + 3 * apal_medio
sospechosos5 <- which(resultados[["covratio"]] < inferior |
                      resultados[["covratio"]] > superior)

cat("- Residuos con razón de covarianza fuera de rango:",
    sospechosos5, "\n")

# Resumen de valores sospechosos.
sospechosos <- c(sospechosos1, sospechosos2, sospechosos3,
                 sospechosos4, sospechosos5)

sospechosos <- sort(unique(sospechosos))

cat("\nResumen de valores sospechosos:\n")
cat("Apalancamiento promedio:", apal_medio, "\n")
```

```r

cat("Intervalo razón de covarianza: [", inferior, "; ",
    superior, "]\n\n", sep = "")

print(round(resultados[sospechosos, c("distancia_Cook", "apalancamiento",
                                      "covratio")], 3))
```

```
Identificación de valores atípicos:
- Residuos estandarizados fuera del 95% esperado: 17 18
- Residuos con una distancia de Cook alta:
- Residuos con apalancamiento fuera de rango: 9 16
- Residuos con DFBeta >= 1: 3 5 6 9 25 28 32
- Residuos con razón de covarianza fuera de rango: 15 16

Resumen de valores sospechosos:
Apalancamiento promedio: 0.125
Intervalo razón de covarianza: [0.625; 1.375]

                    distancia_Cook apalancamiento covratio
Datsun 710                   0.058          0.095     0.919
Hornet Sportabout            0.013          0.093     1.182
Valiant                      0.038          0.097     1.045
Merc 230                     0.162          0.297     1.313
Cadillac Fleetwood           0.008          0.227     1.474
Lincoln Continental          0.001          0.264     1.570
Chrysler Imperial            0.348          0.230     0.724
Fiat 128                     0.146          0.128     0.715
Pontiac Firebird             0.045          0.068     0.856
Lotus Europa                 0.088          0.161     1.050
Volvo 142E                   0.063          0.124     1.016
```

Figura 15.16: identificación de valores atípicos.

15.6.2 Verificación de las condiciones

A fin de que el modelo sea generalizable, tenemos que verificar el cumplimiento de las condiciones descritas en las primeras páginas de este capítulo. Es sencillo comprobar que las variables predictoras son dicotómicas o numéricas a nivel de intervalo y que ninguna de ellas corresponde a una constante. Adicionalmente, las observaciones son independientes entre sí por tratarse de modelos diferentes de automóviles que no parecen seguir un criterio de selección (más que los años de fabricación). A su vez, podemos comprobar que la variable dependiente es numérica a nivel de intervalo sin restricciones.

Al examinar la matriz de correlación (figura 14.4), sin embargo, podemos apreciar una relación positiva moderada entre el tiempo mínimo para recorrer un cuarto de milla y el rendimiento ($R = 0,419$).

La verificación de otras condiciones resulta algo más compleja, por lo que requiere de herramientas matemáticas adicionales.

15.6.2.1 Independencia de los residuos

Esta condición significa que no debe existir autocorrelación en los residuos. Podemos probarlo con una prueba estadística específica conocida con el nombre de sus autores: la prueba de Durbin–Watson, que verifica si dos residuos adyacentes (un retardo), o incluso más alejados, están correlacionados (Durbin & Watson, 1950, 1951). A diferencia de la mayoría de las pruebas de hipótesis, esta prueba define tres regiones: rechazo de H_0, no rechazo de H_0 y una región no concluyente, por lo que existen dos valores críticos, los cuales se buscan en tablas publicadas.

La función `durbinWatsonTest(model)`, del paquete `car`, nos permite aplicar la prueba de Durbin-Watson a los residuos. Sin embargo, debemos tener en cuenta que los resultados de esta prueba dependen del orden de los datos, por lo que al reordenar los datos se podrían obtener resultados diferentes. Al aplicar esta prueba para el ejemplo (script 15.8, línea 12) obtenemos un valor $p = 0,236$, por lo que podemos concluir que los residuos son, en efecto, independientes.

15.6.2.2 Distribución normal de los residuos

Tal como mencionamos previamente, la figura 15.15b muestra que los residuos podrían alejase un poco de la distribución normal. Al aplicar la prueba de Shapiro-Wilk (script 15.8, línea 16), obtenemos como resultado $p = 0,080$, por lo que podemos asumir que el supuesto se cumple, aunque manteniendo cautela por la cercanía con el nivel de significación.

15.6.2.3 Homocedasticidad de los residuos

El gráfico de la figura 15.15a muestra algunos residuos que se alejan levemente del rango general, pero que no parecen ser muy problemáticos.

Una prueba adecuada para verificar esta condición es la de Breusch-Pagan-Godfrey (Glen, 2016), cuya hipótesis nula es que las varianzas de los residuos son iguales. En R, esta prueba está implementada en la función `ncvTest(model)` del paquete `car`. Al usarla para el ejemplo (script 15.8, línea 20), obtenemos como resultado $p = 0,212$, por lo que podemos concluir que el supuesto de homocedasticidad se cumple.

15.6.2.4 Multicolinealidad

Esta condición establece que no debe existir una relación lineal entre dos o más predictores. En otras palabras, la correlación entre variables no debe ser muy alta (o muy baja, si la relación es inversa).

Como hemos dejado entrever a lo largo de este capítulo, el incumplimiento de esta condición puede causar diversos problemas:

- Estimaciones poco confiables de los parámetros. Cuando dos (o más) predictores están perfectamente correlacionados, existen en realidad infinitos valores para sus parámetros que resuelven el problema

de optimización de minimizar la suma de los residuos cuadrados, por lo que la variabilidad de dichos parámetros puede ser muy elevada.
- Limita la magnitud de R, puesto que si los predictores están correlacionados en realidad explican la misma porción de la variabilidad de la respuesta.
- Dificulta evaluar la importancia de cada predictor, por el mismo motivo anterior.

Podemos revisar esta condición mediante el factor de inflación de varianza (VIF) y el estadístico tolerancia ($1/VIF$).

El VIF para cada predictor i se calcula mediante la ecuación 15.8, donde R_i^2 se obtiene usando todos los predictores restantes para ajustar una RLM con el predictor i como respuesta (Frost, 2021).

$$VIF_i = \frac{1}{1 - R_i^2} \tag{15.8}$$

Aunque no hay un acuerdo general, muchos autores usan el valor $VIF \geq 10$ como umbral para preocuparse, aunque hay autores que consideran críticos valores más conservadores, de 5 o incluso de 2,5 (Frost, 2021). También se ha encontrado que si el VIF promedio es mayor a 1, podría haber sesgo en el modelo.

En el caso de la tolerancia, la literatura sugiere que valores bajo 0,2 podrían ser problemáticos, aunque algunos académicos creen que valores cercanos a 0,4 deberían ser revisados.

El paquete `car` de R incluye la función `vif(model)` para calcular el factor de inflación de la varianza. Al usarla para el ejemplo (script 15.8, líneas 23–29) obtenemos los resultados de la figura 15.17.

```
Verificar la multicolinealidad:
- VIFs:
      wt     qsec       am
2.482952 1.364339 2.541437
- Tolerancias:
      wt      qsec        am
0.4027465 0.7329556 0.3934781
- VIF medio: 2.129576
```

Figura 15.17: verificación de condición de multicolinealidad.

Si miramos los factores de inflación de la varianza, en general no parecen ser preocupantes. Sin embargo, los estadísticos de tolerancia son preocupantes. Adicionalmente, el VIF promedio también indica que el modelo podría estar sesgado. Es necesario, entonces, buscar un modelo más confiable. Podríamos, por ejemplo, eliminar la variable menos significativa (`am`, que tiene el menor valor p), obteniendo el modelo con dos predictores de la figura 15.1, el que tendríamos que evaluar y comparar con este modelo de tres predictores (ver los últimos ejercicios propuestos).

Script 15.8: iverificación de condiciones para el modelo.

```
1  library(car)
2
3  # Cargar datos.
4  datos <- mtcars
5
6  # Ajustar modelo.
7  modelo <- lm(mpg ~ wt + qsec + am, data = datos)
8
9  # Comprobar independencia de los residuos.
10 cat("Prueba de Durbin-Watson para autocorrelaciones ")
11 cat("entre errores:\n")
12 print(durbinWatsonTest(modelo))
13
```

```r
# Comprobar normalidad de los residuos.
cat("\nPrueba de normalidad para los residuos:\n")
print(shapiro.test(modelo$residuals))

# Comprobar homocedasticidad de los residuos.
cat("Prueba de homocedasticidad para los residuos:\n")
print(ncvTest(modelo))

# Comprobar la multicolinealidad.
vifs <- vif(modelo)
cat("\nVerificar la multicolinealidad:\n")
cat("- VIFs:\n")
print(vifs)
cat("- Tolerancias:\n")
print(1 / vifs)
cat("- VIF medio:", mean(vifs), "\n")
```

15.6.3 Validación cruzada

Al igual que en el caso de regresión lineal simple, también podemos usar validación cruzada como herramienta para mejorar la estimación del error cuadrado medio. No hay diferencia en la realización de este proceso con respecto a lo estudiado en el capítulo anterior, por lo que no se aborda aquí con mayor detalle.

15.6.4 Tamaño de la muestra

Otro aspecto importante a tener en cuenta al momento de determinar si un modelo de RLM es confiable es el tamaño de la muestra. Existen distintas reglas que simplifican la tarea de determinar el tamaño de la muestra, pero lo cierto es que mientras más observaciones tengamos, mejor. Una de las reglas más simplistas en verificar que se tengan al menos 10 o 15 observaciones por cada predictor. De acuerdo a este criterio, la muestra del ejemplo cuenta con 32 observaciones, con 3 variables predictoras en el modelo, por lo que estaríamos en el borde inferior recomendado.

Considerando, además, que ya hemos encontrado indicios de que el modelo de tres variables podría no ser confiable, parece ser más pertinente usar un modelo más sencillo, como el de la figura 15.1.

15.7 EJERCICIOS PROPUESTOS

1. ¿En qué se diferencia la regresión lineal simple (RLS) de la regresión lineal multivariada (RLM)?
2. ¿Cómo luce la ecuación de un modelo RLM?
3. Si un modelo RLS y un modelo RLM usan una misma variable, ¿tendrán los mismos coeficientes?
4. ¿Qué son los predictores colineales y por qué hay que identificarlos?
5. Explica qué es el coeficiente de determinación R^2 ajustado y para qué se usa.

6. Explica qué son los modelos nulo y completo en el contexto RML.
7. Explica en qué consiste la estrategia de selección de variables hacia adelante.
8. Explica en qué consiste la estrategia de eliminación de variables hacia atrás.
9. Enumera las condiciones necesarias para aplicar RLM.
10. Explica qué se puede ver en los gráficos que entrega la función `plot(modelo)` cuando `modelo` es una RLM.
11. Verifica si el modelo presentado en la figura 15.1 cumple las condiciones para poder usar RLM.
12. ¿Son significativamente distintos los modelos de las figuras 15.1 y 15.12?

REFERENCIAS

Diez, D., Barr, C. D. & Çetinkaya-Rundel, M. (2017). *OpenIntro Statistics* (3.ª ed.). https://www.openintro.org/book/os/.

Durbin, J. & Watson, G. S. (1950). Testing for serial correlation in least squares regression: I. *Biometrika*, *37*(3/4), 409-428.

Durbin, J. & Watson, G. S. (1951). Testing for serial correlation in least squares regression: II. *Biometrika*, *38*(1/2), 159-179.

Field, A., Miles, J. & Field, Z. (2012). *Discovering statistics using R*. SAGE Publications Ltd.

Frost, J. (2021). *Variance Inflation Factors (VIFs)*. Consultado el 17 de junio de 2021, desde https://statisticsbyjim.com/regression/variance-inflation-factors/

Glen, S. (2016). *Breusch-Pagan-Godfrey Test: Definition*. Consultado el 16 de junio de 2021, desde https://www.statisticshowto.com/breusch-pagan-godfrey-test/

Montero Muñoz, J., Solla Suárez, P. E. & Gutiérrez Rodríguez, J. (2012). Estimación del filtrado glomerular en el paciente anciano. Implicaciones clínicas en el uso de antibióticos. *Revista Española de Geriatría y Gerontología*, *56*(5), 268-271-.